Goal-driven Automation of a Deep Space Communications Station: A Case Study in Knowledge Engineering for Plan Generation and Execution

Randall W. Hill, Jr., Steve A. Chien, and Kristina V. Fayyad

Jet Propulsion laboratory, California institute of Technology

Contact Author:

Steve Chien

4800 Oak Grove Drive, M/S 525-3660

Pasadena, CA 91109-8099

USA

Email: steve.chien@jpl.nasa.gov

Voice: +1(818) 30(1-6144 FAX: +1 (818) 306-6912

Keywords: Artificial Intelligence, Planning and Scheduling, Architecture

Goal-driven Automation of a Deep Space Communications Station: A Case Study in Knowledge Engineering for Plan Generation and Execution

Abstract

This paper describes the the application of Artificial Intelligence techniques for plan generation, plan execution, and plan monitoring to automate a Deep Space Communication Station. This automation allows a Communication station to respond to a set of tracking goals by appropriately reconfiguring the communications hardware and software to provide the requested communications services. In particular this paper describes: (I) the overall automation architecture, (2) the plan generation and execution monitoring Al technologies used and implemented software components, and (3) the knowledge engineering process and effort required for automation. This automation was demonstrated in Febuary 1995, at the 1> S-13 Antenna Station in Goldstone, CA on a series of Voyager tracks and the technologies demonstrated are being transferred to the operational Deep Space Network stations.'

1.0 Introduction

The Deep Space Network (DSN) [1] was established in 1958 and since then it has evolved into the largest and most sensitive scientific telecommunications and radio navigation network in the world. The purpose of the

---

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was conducted as part of the DSN q'ethnology Program managed by Dr. Chad Edwards.

For further information contact: steve.chien@jpl.nasa.gov.

Third, we describe in detail the knowledge engineering effort required to automate the DSS-13 Deep Space Station.

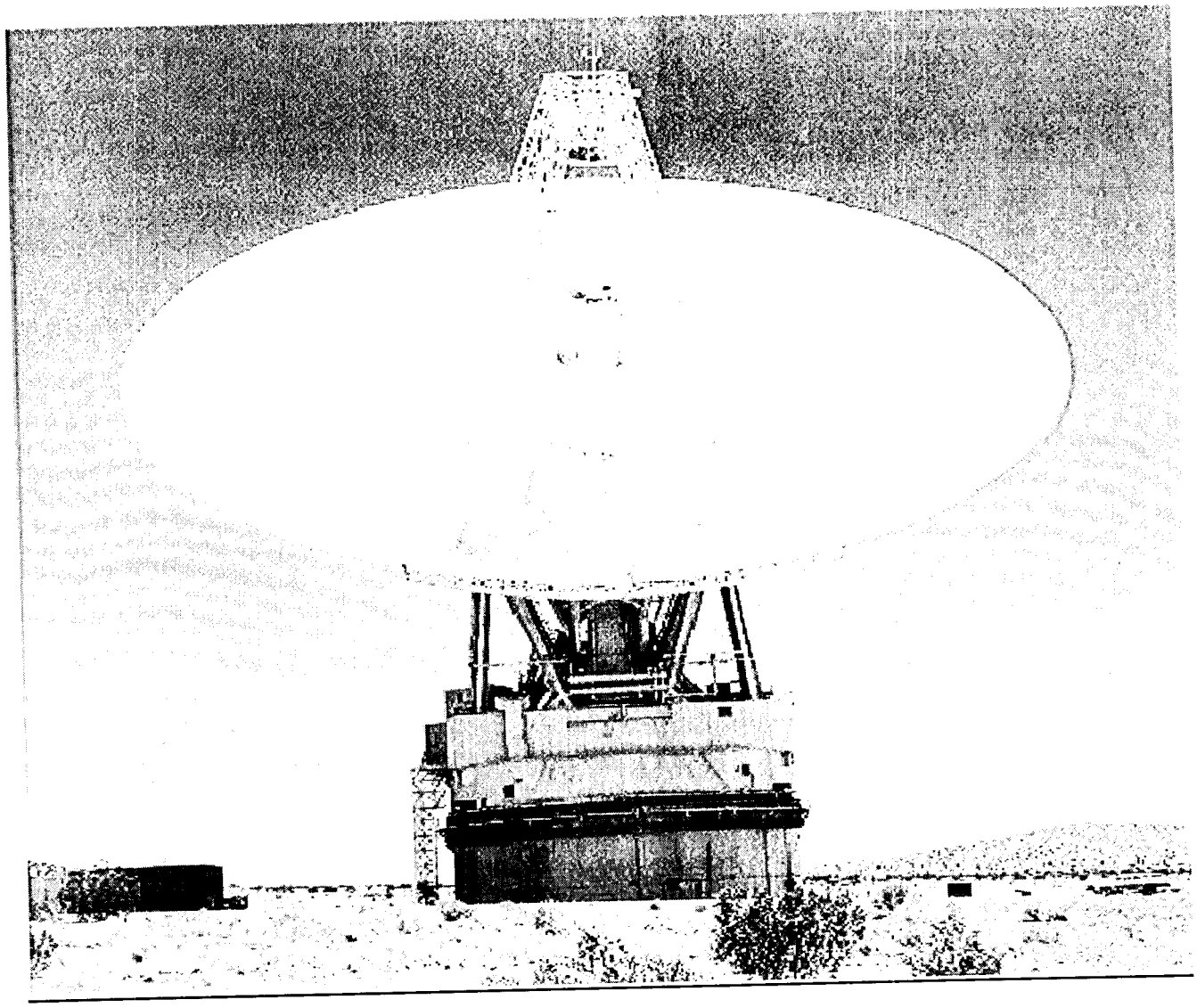


Figure O: A 70-meter Deep Space Network Antenna located at Goldstone, CA.

This paper is organized in the following manner. We begin by characterizing the operation of the DSN at the time that this research was performed. Next we describe the high-level architecture envisioned to automate operations of the Deep Space Network (DSN). In this section we give a functional description of each of the components, which includes the OMP scheduling system for automated resource allocation, DPLAN, an auto-

DSN is to support unpiloted interplanetary spacecraft missions and support radio and radar astronomy observations in the exploration] of the solar system and the universe. '1'here are three deep space communications complexes, located in Canberra, Australia, Madrid, Spain, and Goldstone, California. Each DSN complex operates four deep space stations -- one 70-meter antenna, two 34-meter antennas, and one 26-meter antenna 1. The functions of the DSN are to receive telemetry signals from spacecraft, transmit commands that control the spacecraft operating modes, generate the radio navigation data used to locate and guide the spacecraft to its destination, and acquire flight radio science, radio and radar astronomy, very long baseline interferometry, and geodynamics measurements. Figure 0 shows a picture of a Deep Space Network 70-meter antenna located at Goldstone, CA.

From its inception the DSN has been driven by the aced to create increasingly more sensitive telecommunications devices and better techniques for navigation. The operation of the DSN communications complexes require a high level of manual interaction with the devices in the communications link with the spacecraft. In more recent times NASA has added some new drivers to the development of the DSN: (I) reduce the cost of operating the DSN, (2) improve the operability, reliability, and maintainability of the DSN, and (3) prepare for a new era of space exploration with the New Millennium program: support numerous small, intelligent spacecraft while Using very few mission operat ions personnel.

The purpose of this paper is threefold. First, we describe an architecture for automating a Deep Space Station to allow it to fulfill goal-based requests to capture spacecraft data. In particular, we will describe how the components of the architecture transform a flight project service request into an executable set of DSN operations that fulfill the request through automated resource allocation, goal-driven plan generation, and plan execution and monitoring. Second, we describe how each of the consituent Al technologies of plan generation, and plan execution monitoring were used to enable goal-driven automation of the DSS-13 Deep Space Station.

---

[1]Each one of these antenna stations is called a Deep Space Station, or DSS. Individual Deep Space Stations at-c numbered to distinguish them, for example DSS-13 and 1) SS-28 are stations at Goldstone, CA.

Center located at JPL. The entire process is initiated when a flight project sends a request for the DSN to track a spacecraft. This request specifics the timing constraints of the track (e.g., when the spacecraft can be tracked), data rates, frequencies required, as well as the services required (i. e., downlink of information from the spacecraft, commanding uplink to the spacecraft, etc.). The DSN responds to the request by performing a process called Network Preparation. The Network Preparation Process includes attempting to schedule the resources (i.e., an antenna and other required subsystems such as receivers, exciters, telemetry processors) needed for the track as well as generating necessary data products required to perform the track (predictions of the spacecraft location relative to the ground station, transmission frequencies, etc.). The output of this process is a schedule of tracks to be performed by DSN groundstations, equipment allocations to tracks, and supporting data required for tracks. One key part of this supporting data the Sequence of Events (SOE) describing the time-ordered activities that should occur during the track. The SOE includes actions that the DSN should take, (e.g., begin tracking the project's spacecraft at 1200 hours), and it also includes events that will occur on the spacecraft being tracked (e.g., the spacecraft will change frequency or mode at a designated time and the DSN should anticipate the event). Additionally, the DSN must generate predict information required for the track. This is information about where in the sky the spacecraft will be relative to the antenna so that the antenna can be directed to the correct orientation to acquire the spacecraft and to maintain pointing daring the track as the earth rotates and moves and the spacecraft moves.

mated procedure generation system, and LMCOA, a plan execution anti monitoring system. In addition we provide examples of the inputs and outputs to each of the components to illustrate what occurs at each step in the process of capturing spacecraft data. Next, we focus on the two components of the system involved in automation of a single Deep Space Station: the DPLAN planning system and the LMCOA plan execution anti monitoring system. Specifically we describe the knowledge representation used for DPLAN's decomposition rules and for the resultant temporal dependency networks (TDN's) used by LMCOA. Next, we describe the knowledge acquisition and validation processes performed while building the automation system. Finally, we describe the results of the technology demonstration at 1) S-13 and ongoing efforts to insert the demonstrated AI technology into the operational DSN.

2.0 Current DSN Operations

Voyager- 1 is cruising at 17.5 kilometers/second toward the outer edge of the solar system. Though its onboard systems are mostly asleep during this phase of its mission, Voyager's health metrics are continually sent to Earth via a telemetry signal radiated by its 40-watt transmitter. It will take eight hours at the speed of light for the signal to reach its destination, Earth, a billion miles away. Upon arrival, the telemetry signal is received by a Deep Space Station which is part of an extremely sensitive ground communications system, NASA's Deep Space Network (DSN). At this station the signal is recorded, processed, and sent to the Mission operations and Voyager project engineers at JPL, who assess the health of the spacecraft based on the contents of the signal.

The type of activity just described occurs daily for dozens of different NASA spacecraft and projects that use the DSN to capture spacecraft data. Though the process of sending signals from a spacecraft to Earth is conceptually simple, in reality there are many earthside challenges that must be addressed before a spacecraft's signal is acquired and transformed into useful information.

Network Preparation at the Network Operations Control Center

Figure 1 shows a simplified depiction of DSN operations (see [2] for a more complete description of the DSN processes). The first stage is called Network Preparation and it occurs at the Network operations Control

erators continually monitor the status of the link and handle exceptions as they occur. For example, the ground station may lose the spacecraft signal (this occurrence is called "the receiver breaking lock with the spacecraft"). In this case the operations personnel must take immediate action to reacquire the spacecraft signal as quickly as possible to minimize the amount of data lost. All of these actions are currently performed by human operators, who manually issue tens or hundreds commands via a computer keyboard to the link subsystems. The monitoring activities require the operator to track the state of each of the subsystems in the link (usually three to five subsystems), where each subsystem has many different state variables that change over tire.

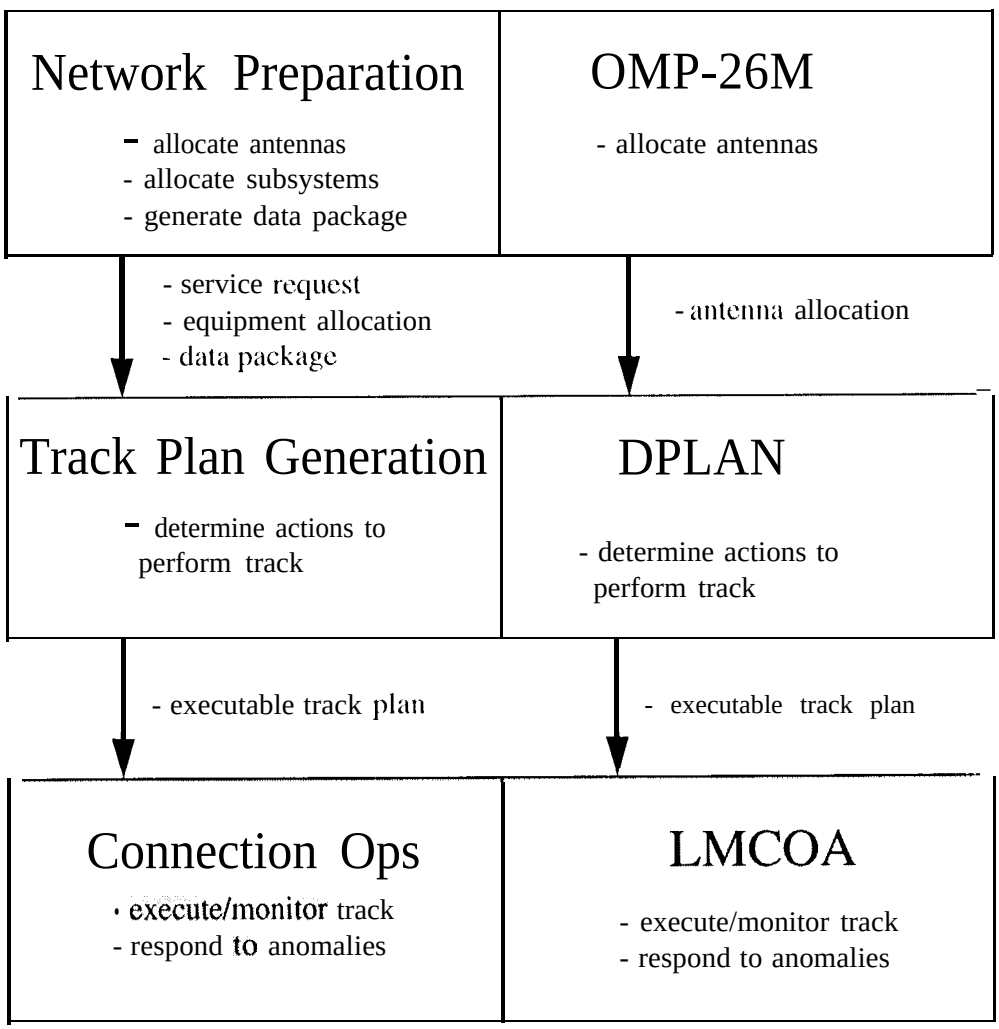


Figure 2. Current Prototype Systems for DSN Automation

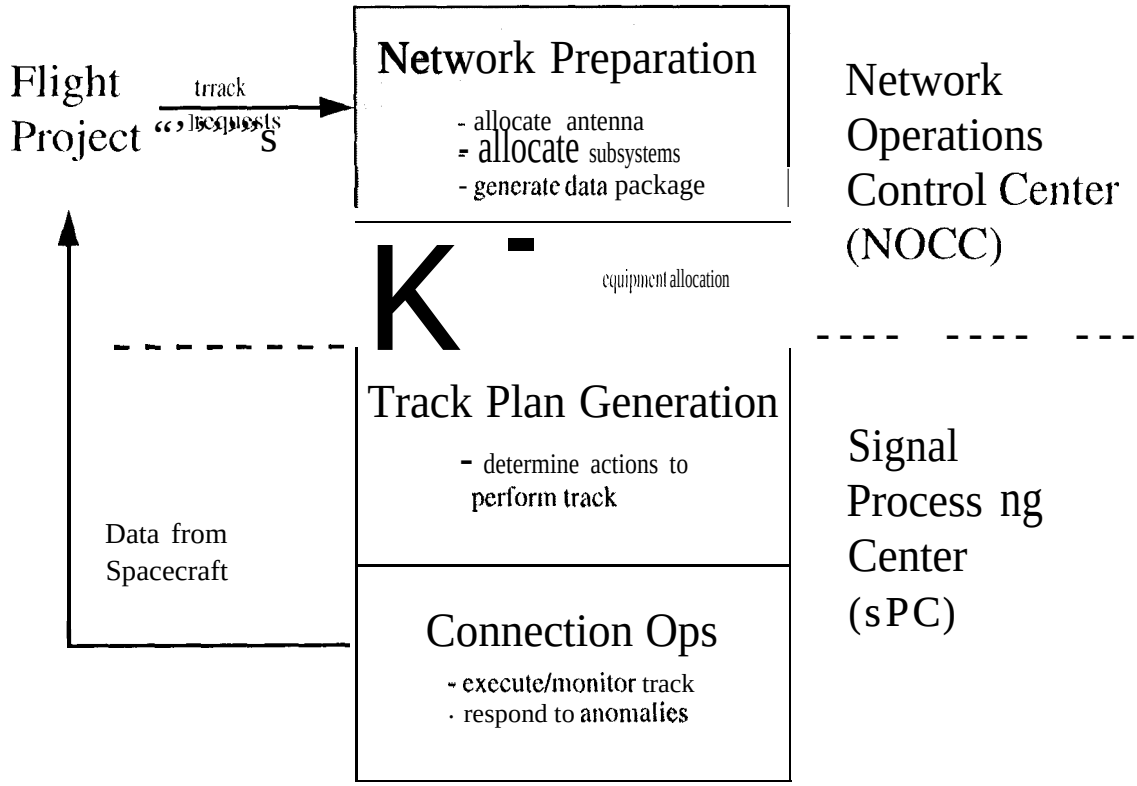


Figure 1: An Overview of DSN Operations

Data Capture at the Signal Processing Center

Once the schedule has been determined, and the SOE and predict information generated, the DSN operations process moves to the Signal Processing Centers (SPC)[2] where a process called data capture occurs. The data capture process is performed by operations personnel at the deep space station. First they determine the correct operations necessary to perform the track. Then they perform these actions--they configure the equipment for the track, establish the communications link, which we hereafter refer to as a 'link', and then perform the track by issuing control commands to the various subsystems comprising the link. Throughout the track the op-

[2] To explain in further detail, in the operational DSN Deep Space Stations (DSSs) are organized into complexes where several DSSs share a pool of common subsystems. These complexes are calles Signal Processing C'enters (S PCs). However, the prototyping work described in this paper took place at the DSN's reseat-ch station DSS-13. DSS-13 dots not share subsystems with other DSSs because its equipment tends to bc different (C. g.) experimental or test versions) and hence does not belong to a SPC. Thus in the operational DSN the trackplan generation and connection operations efforts would be at an SPC, but for this work they took place at a DSS. From an Al research standpoint the reader can assume that SPCs and DSSs at-c interchangeable.

The automated track procedure generation problem involves taking a general service request (such as telemetry - downlink of data from a spacecraft) and an actual equipment assignment (describing the type of antenna, receiver, telemetry processor, and so on), and generating the appropriate partially ordered sequence of commands (called a Temporal Dependency Network or TDN) for creating a communications link to enable the appropriate interaction with the spacecraft. The DSN Antenna Operations Planner (DPLAN) uses an integration of AI Hierarchical Task Network (HTN) [4] and partial order operator-based planning techniques [5] to represent DSN antenna operations knowledge and to antenna operations procedures on demand from the service request and equipment assignment.

LMCOA: Automated Procedure Execution

The automated execution component, called the Link Monitor and Control Operator Assistant (LMCOA) uses the TDN (Figure 4) generated by DPLAN to perform the actual track and is responsible for monitoring the execution of the TDN[3]. This involves ensuring that the expected conditions and subsystem states are achieved, certain types of closed-loop control and error recovery are performed in a timely fashion, and the correct dispatching of commands to the subsystems controlling the link occurs. LMCOA uses an operator-based representation of the TDN to represent necessary and desired conditions for execution of procedures and tracks relevant subsystem state.

Thus, the combination of OMP, DPLAN, and LMCOA enables automation of a significant portion of DSN operations. In the remainder of this article, we focus on the DPLAN and LMCOA systems and how they combine to allow goal-driven automation of a Deep Space Communication Station.

4.0 DPLAN: Automated Procedure Generation

[3] LMCOA also uses the predict and SOE information generated by DSN operations to execute the track. Automatic generation of this essential information is an important portion of automating DSN operations but falls outside the scope of DPLAN. There are currently other efforts within the DSN to automate these processes relating to the generation of these types of information.

3.0 An Architecture for Automation of the DSN

In the last section we described the current process for transforming a flight project service request into an executable set of DSN operations. As we have already pointed out, many of the steps of the described processes are knowledge and labor intensive. This paper describes efforts to automate portions of the process map shown in Figure I using Artificial intelligence technologies. The specific tools and how they map onto the current DSN functions is shown below in Figure 2. Specifically, the Operations Mission Planner 26-Meter (OMP-26M) system is applied to the resource scheduling process, the Deep Space Network

Antenna Operations Planner (DPLAN) is used for automatically generating DSN operations procedures, and the Link Monitor and Control Operator Assistant (1 .MCOA) automatically executes the operations procedures and performs connection operations.

OMP-26M: Automated Scheduling

The high level resource allocation problem for the DSN is handled by the Operations Mission Planner (OMP-26M) [1 1] scheduling system. The OMP system accepts as inputs: (I) generalized service requests from spacecraft projects of the form "we need three 4-hour tracks per week" and (2) specific requests of the form "we need to perform maintenance on 1) SS-28 from I ()()0- 1600 on Wednesday October 24th. OMP then produces a specific schedule allocating resources to requests, resolving conflicts using a priority request scheme which attempts to maximize satisfaction of high priority projects. This automated scheduling and resource allocation function corresponds to the processes previously described as "schedule resources" and "resource management." OMP deals with schedules for NASA's 2(J-meter subnet involving thousands of possible tracks and a final schedule involving hundreds of tracks. While OMP performs a vital function in the automation architecture, this paper focuses on the elements of automation which were used to automate a single Deep Space Station, rather than resource allocation of a network of stations. Thus we will primarily discuss the other two components, DPLAN and LMCOA.

1)1'1 .AN: Automated Procedure Generation

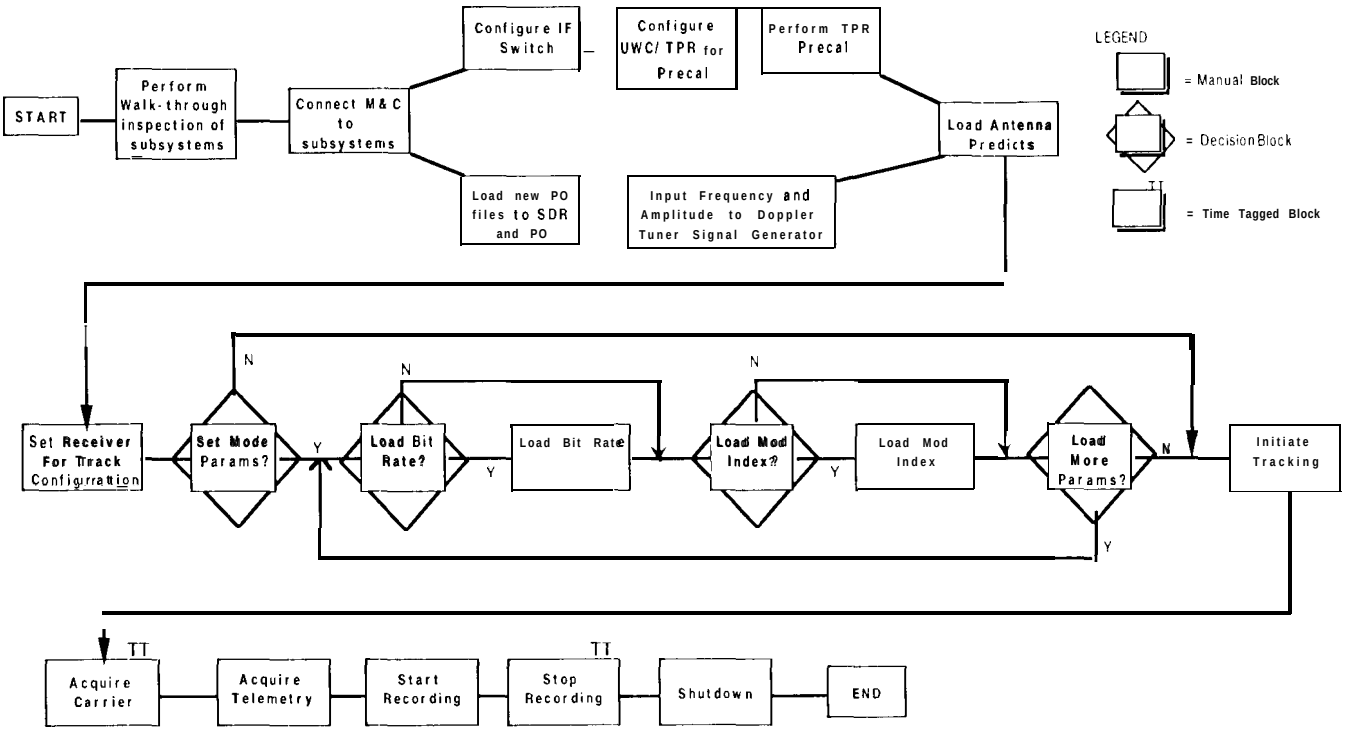


Figure 4: Temporal Dependency Network for Voyager Track

4.1 Track I'Ian Generation: inputs and Outputs

The DPLAN planner uses high level track information to clctc'mine appropriate steps, ordering constraints on these steps, parameters of these steps to achieve the high level track goals given the equipment allocation. in generating the TDN, the planner uses information from several sources (see Figure 3):

Service Request: The service request specifies the goals of the track (e. g., to provide certain DSN services over a specified period of time). These include downlink/telemetry, commanding, ranging, and radio science types of services. A sample set of service requests/tracking goals is shown in Figure 5.

Project SOE: The project sequence of events specifies events from the mission/project perspective. Relevant information specified in the project SOE includes such items as the one-way light time (OWLT) to the spacecraft, notifications of the beginning and ending times of tracks, spacecraft data transmission bit rate changes, modulation index changes, and carrier and subcarrier frequency changes.

As stated above, the automated track procedure generation problem involves taking a general service request and an equipment assignment and generating a plan to run the track (called a Temporal Dependency Network or TDN; see Figure 3) for creating a communications link to enable the appropriate interaction with the spacecraft. The DSN Antenna Operations Planner (DPLAN) uses an integration of Al Hierarchical Task Network (HTN) [4] and partial order operator-based planning techniques [5] to represent DSN antenna operations knowledge and to antenna operations procedures on demand from the service request and equipment assignment. In this section wc first describe the inputs and outputs of the DPLAN system. Wc then describe how DPLAN represents knowledge of DSN antenna operations procedures and constraints.

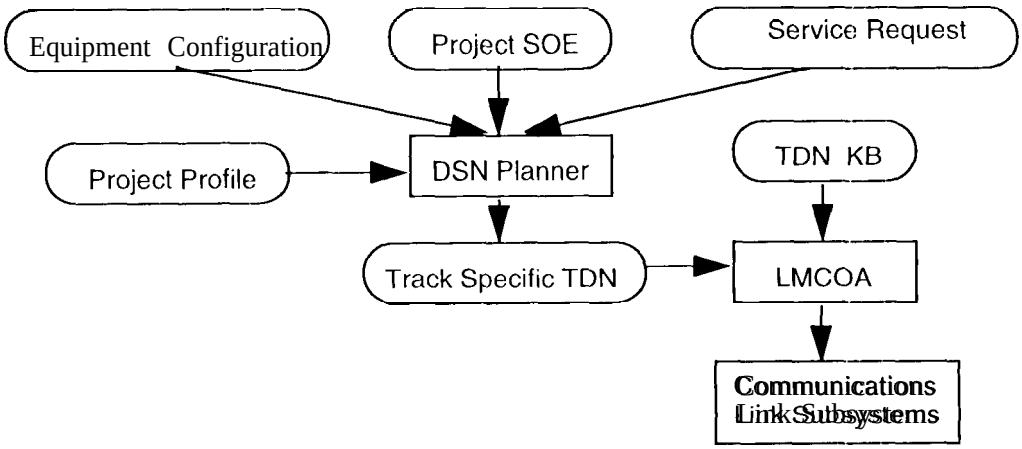


Figure 3: DPLAN and LMCOA inputs and Outputs

chy and equipment goal hierarchy, where the rule applies to all contexts below the rule in the relevant hierarchy (all specializations of its scope).

Using this problem specification, the DSN planner then uses task reduction planning techniques (also called hierarchical task network or HTN) [4] and operator-bawd planning techniques [5] to produce a parameterized track-specific TDN to be used to conduct the track. The actual planner used to generate the TDN is a modified version of the task reduction planning component of the Multimission VICAR Planner system [6]. This track-specific TDN anti the SOE can then be used by LMCOA to operate the actual antenna to conduct the requested antenna track.

DPLAN uses a hierarchical knowledge representation to represent DSN operations procedures. This allows a knowledge engineer to specify the scope of each piece of knowledge (i. e., to which set of goals or equipment types to which a rule or constraint applies). For example, Figure 6 below shows a partial tracking goal hierarchy involving the goals telemetry, commanding, and ranging. Figure 7 below shows a partial track hierarchy for antennas. A ruie might specify how to achieve a tracking goal fora particular type of antennas. For example, a ruie might specify how to achieve the telemetry tracking goal for a 34m BWG antenna. Alternatively, a ruie might specify a constraint on how achieving telemetry might be constrained for all HEF antennas. For example, a rule might specify that two receiver calibration steps A and B might have to be ordered so that A is always before B. By representing the track, equipment, and other hierarchies the scope of various pieces of knowledge regarding antenna track activities can be naturally and easily represented.

Project profile: This file specifics project specific information regarding frequencies and pass types. For example, the Project SOE might specify frequency = HIGH, and the project profile would specify the exact frequency used. The project profile might also other signal parameters and default track types.

TDN **KB:** The Temporal Dependency Network (TDN) knowledge base [3] stores information on the TDN blocks available for the DSN Planner and LMCOA to use. This knowledge base includes information regarding preconditions, postconditions, directives, and other aspects of the TDN blocks. It also includes information on how to expand the block parameters and name into the actual flatfile entry in a TDN.

Equipment Configuration: This details the types of equipment available and unique identifiers to be used to specify the exact pieces of equipment to be used in the track. These include the antenna, antenna controller, the receiver, and so on.

```
(PO_required track 1 )
(spacecraft_mode_changes track 1 )
(track_goal spacecraft_ track telemetry track I )
(track_goal spacecraft_track downlink track I )
(track_goal decode_data)
(station-used track 1 DSS I 3)
```

Figure 5: Example Tracking Goals

4.2 How DPLAN Constructs Tracking Plans

I) PLAN uses the tracking goals which comprise part of the SOE to generate the operations procedure for a track (see Figure 5 for sample tracking goals). The DSN planner reduces the high level track goals into executable steps by applying knowledge about how to achieve specific combinations of track goals in the context of specific equipment combinations. This information is represented in the form of task reduction rules, which detail how a set of high level goals can be reduced into a set of lower level goals in a particular problem-solving context. Each task reduction rule rigorously details the scope of its expertise in terms of track and equipment combinations. The information of scope of applicability of the rule can be considered in terms of a track goal hierar-

For example, Figure 8 shows a graphic summary of a task reduction rule. This rule has two context conditions: that the station being operated is DSS-13 and that tracking goal "downlink track" be present. The rule states that if these context conditions are met, the abstract task of pre-calibration can be achieved by performing the lower level tasks of inspecting the subsystems, connecting the subsystems, configuring the total power radiometer, loading antenna predicts, and configuring the receiver. Furthermore, the subsystems must be inspected before connecting the subsystems, and so on. Note that some of these tasks are not operational tasks and will be later expanded into more detailed (operational) tasks. For example, configuring the total power radiometer involves configuring the IF switch, configuring the UWC/TPR for pre-calibration, and performing the actual TPR pre-calibration.

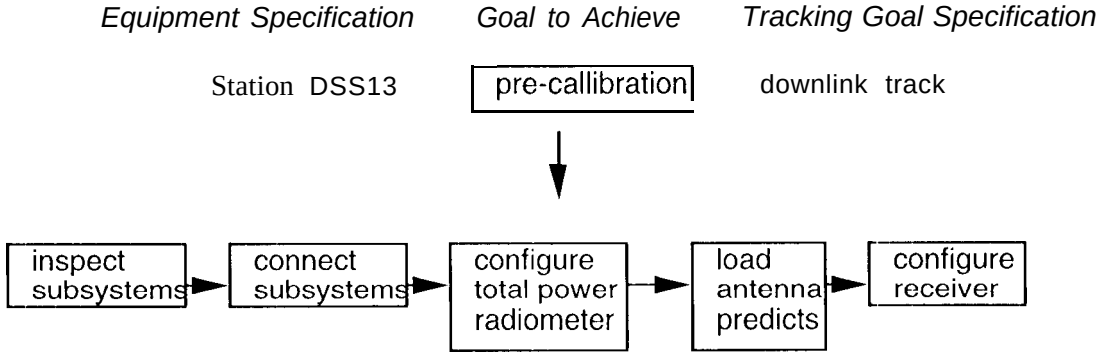
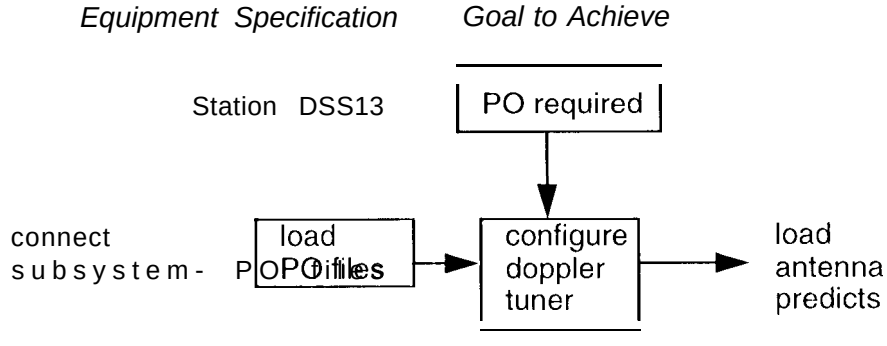


Figure 8: Sample Task Reduction

Next consider the task reduction rule shown in Figure 9. It states that in the equipment context of 1) SS13 and in any tracking goal context, the abstract goal of including a programmable oscillator (PO) can be achieved by adding the steps: load PO files followed by configure doppler tuner. Additionally, these steps must be ordered with respect to connect subsystems and load antenna predicts steps as indicated



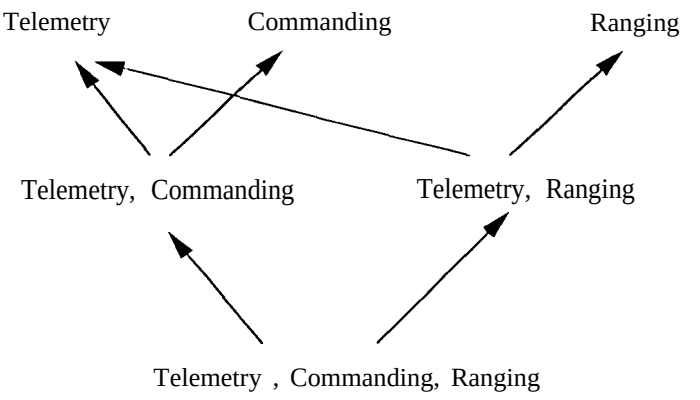


Figure 6: Partial Track Hierarchy

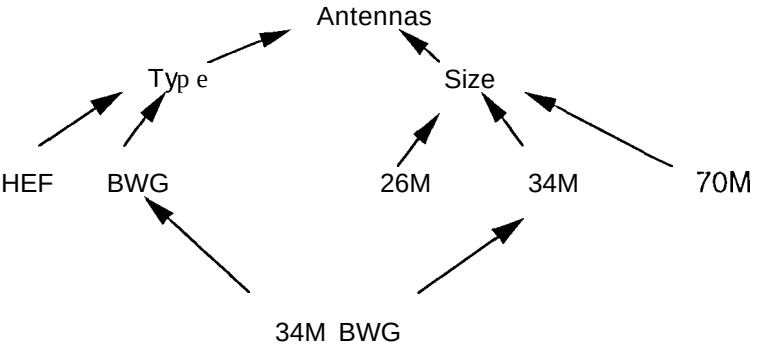


Figure 7: Example Equipment 1 hierarchy - Antenna Types

Using this problem specification, the DSN planner then uses task reduction planning techniques (also called hierarchical task network or }] TN) [4] and operator-based planning techniques [5] to produce a parameterized track-specific TDN to be used to conduct the track. The actual planner used to generate the TDN is a modified version of the task reduction planning component of the Multimission VICAR Planner system [6]. This track-specific TDN and other elements of DSN support data are then be used by LMCOA to operate the actual antenna to conduct the requested antenna track.

in HTN Planning, task redaction rules specify how to rt'date abstract activities into lower level activities. This process continues until all of the activities in a plan have been reduced into operational (i.e., executable) activities. In a task redaction rule, one specifics a goal G to be reduced, a context set of conditions C (which restrict the cases in which the rule applies), a set of lower level goals L (i.e., G is being reduced into 1.), and a set of constraints O, which specify constraints on the new goals L.

Ka-band		*	2
Q-band		*	2
S&X-band	*	*	3
X&Ka-band	*	*	3

Figure 11: TPR IF Switch Parameter Determination

The application of decomposition rules continues until all of the activity blocks in the TDN are operational - that is to say that known blocks in the TDN KB can be used to instantiate the each and every activity in the TDN. This fully instantiated TDN can then be used with the LSOE by LMCOA to perform the actual track. Shown above in Figure 3 is the TDN for a VOYAGER downlink telemetry track using the Programmable oscillator for the DSS13 DSN antenna station.

5.0 LMCOA: Automated Procedure Execution

The automated execution component, called the Link Monitor and Control Operator Assistant (LMCOA), uses the TDN (Figure 3) generated by DPLAN to perform the actual track. LMCOA is responsible for monitoring the execution of the TDN. This involves ensuring that the expected conditions and subsystem states are achieved, certain types of closed-loop control and error recovery are performed in a timely fashion, and the correct dispatching of commands to the subsystems controlling the link occurs. LMCOA uses an operator-based representation of the TDN to represent necessary and desired conditions for execution of procedures and tracks relevant subsystem state.

The LMCOA performs the operations procedures for a tracking activity by executing a Temporal Dependency Network (TDN), which is a procedure that is automatically generated by DPLAN, as described in the last section. DPLAN composes the TDN so that it contains the procedures (TDN blocks) needed for a specific tracking activity, and it orders them according to its knowledge of the dependencies that are defined among the blocks as well as by what it knows about the pre- and postconditions of the blocks. The knowledge about interlock dependencies and about block pre- and postconditions is passed to the LMCOA, whose task it is to execute the end-to-end procedure. The LMCOA receives the TDN in the form of a directed graph, where the precedence

Figure 9: Augmentation of TDN Required by Additional Programmable Oscillator Track Goal

In the context of specific tracks, generic configuration blocks will be specialized as appropriate to the details of the track. For example, the task reduction rule in Figure 10 states that in the equipment context of Station DSS13, if the track context is downlink, telemetry, with symbol decoding requested, "receiver configuration block type A" is the appropriate one to use to configure the receiver.

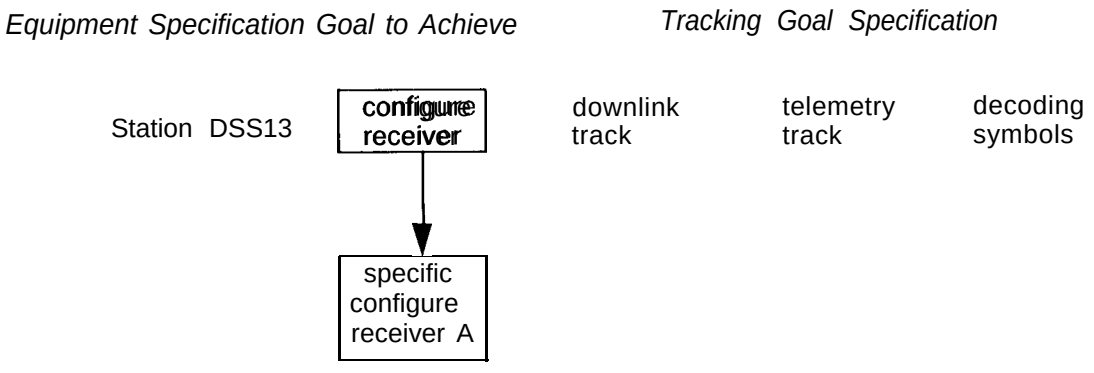


Figure 10: A Specialized Task Reduction

Considerable effort in computing the final TDN is devoted to determining the correct parameters for blocks in the TDN. For example, Figure 11 shows a configuration) table used to determine the IF switch parameter for the TPR precalibration step. Depending on the communication bands used in the track, differing bands will be assigned to each of the communication pathways in the UWC/TPR. Based on the bands being used, the TPR IF switch parameter is also determined. This parameter setting is also determined during the decomposition process, so a correctly parameterized TDN can be constructed. In this case the tabular information is represented by several rules each with context conditions corresponding to the bands used and each rule specifying the correct IF1, 1F2, and T1'R IF Switch Parameter settings.

Band	IIF1	IF2	Parameter
S-band	*		
X-band	*		

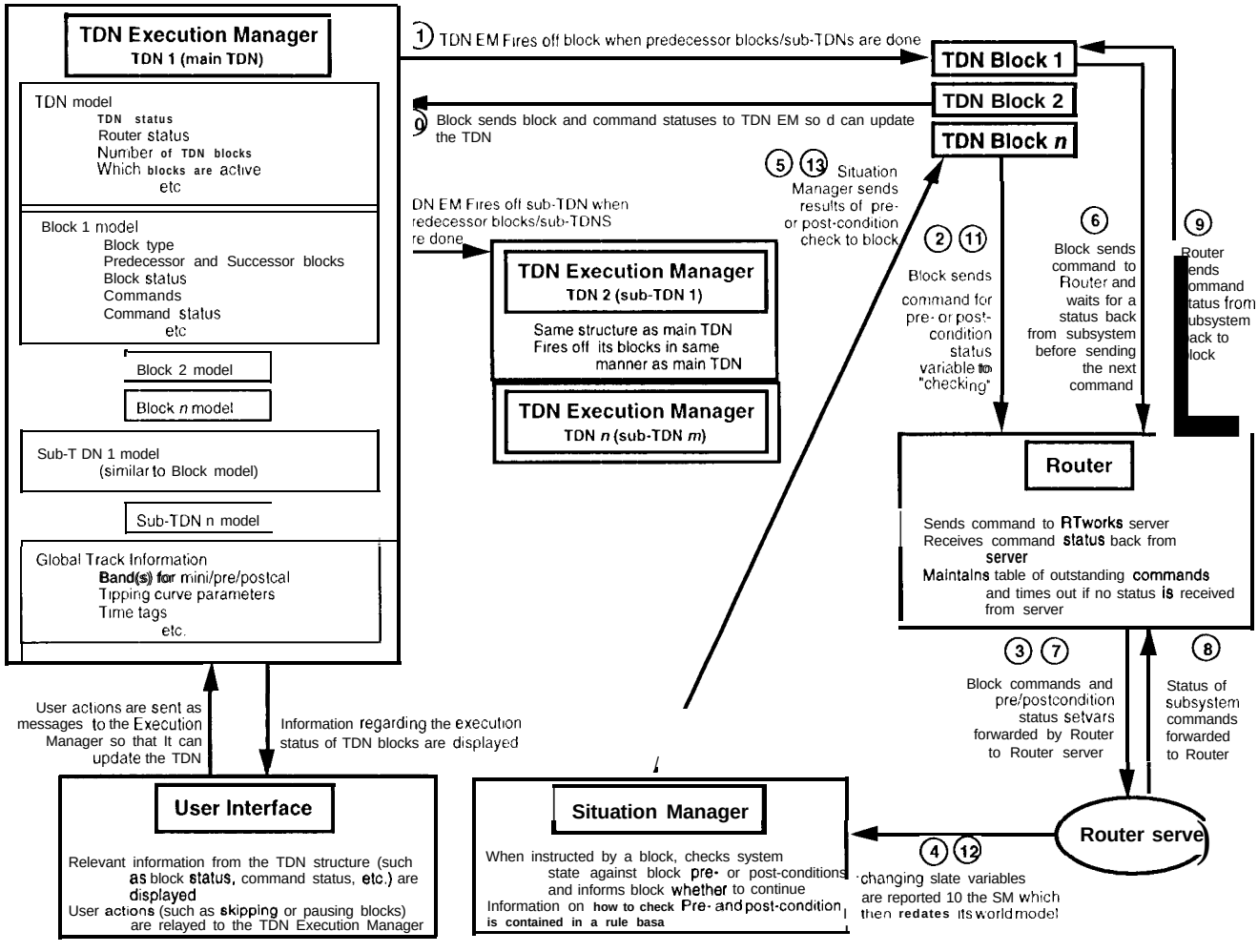


Figure 12: LMCOA Architecture

Once a TDN block begins to execute, it requests the Situation Manager, via the Router and the Rtserver, to verify that the block's preconditions are satisfied. A precondition is a state that must be true in the physical world. For example, in Figure 4 the block named "Configure IF Switch" has a precondition that the connection between the M&C system and the subsystems has been established. 'T'bough this condition is the intended effect of the previous block, the precondition check verifies that the desired state actually holds prior to attempting to configure the IF switch. The result of the precondition check is sent back to the block, and the block forwards this information to the EM, which keeps track of the status of each block.

Once the Situation Manager (SM) verifies that the preconditions are met, the block then dispatches its commands, one at a time, through the Router and the Rrl'server, to the subsystems it is controlling. When a sub-

relations arc specified by the nodes and arcs of the network. The blocks in the graph are partially ordered, meaning that some blocks may be executed in parallel. Temporal knowledge is also encoded in the TDN, which includes both absolute (e.g. Acquire the. spacecraft at time 02:30:45) and relative (e.g. Perform step Y 5 minutes after step X) temporal constraints. Conditional branches in the network are performed only under certain conditions. Optional paths are those which are not essential to the operation, but may, for example, provide a higher level of confidence in the data resulting from a plan if performed. More details about TDNs are provided in [3].

To execute a TDN, LMCOA performs the following functions: (I) it loads the parameterized TDN into the Execution Manager; (2) it determines which TDN blocks are eligible for execution and spawns a process for each TDN block; (3) it checks whether the preconditions of each TDN block have been satisfied; (4) once the preconditions are satisfied, it issues the TDN block commands; and (5) it verifies whether the commands had their intended effects on the equipment. The Operator interacts with the LMCOA by watching the execution of the TDN; the operator can pause or skip portions of the TDN, check the status of commands within individual blocks, and provide inputs where they are required. When part of a TDN fails, the LMCOA supports manual recovery by the operator by highlighting the point of failure and providing information about the preconditions or postconditions that failed to be sat i sfied.

The LMCOA architecture is shown in Figure 12. The Execution Manager (EM) oversees the overall execution of the TDN: it loads the TDN, selects the TDN blocks that are eligible for execution, spawns a thread process for each of the blocks that are ready for execution (shown as TDN Block 1, TDN Block 2, and TDN Block n in Figure 12), and monitors the execution status of the blocks. The EM selects a block for execution when all of that block's predecessor have been successfully executed. Since many of the TDN blocks require track-specific parameters, the EM finds the parameters in the Sequence of Events (SOE) information.

requites about **900** operator inputs overall, if the track is performed manually. For this same track, under nominal conditions, LMCOA reduces the number of operator inputs to less than 10. Since humans tend to be more error prone than computers on simple repetitive tasks, it makes sense to assign these tasks to LMCOA, freeing the operator for the task of handling exceptions, which requires more intelligence and knowledge.

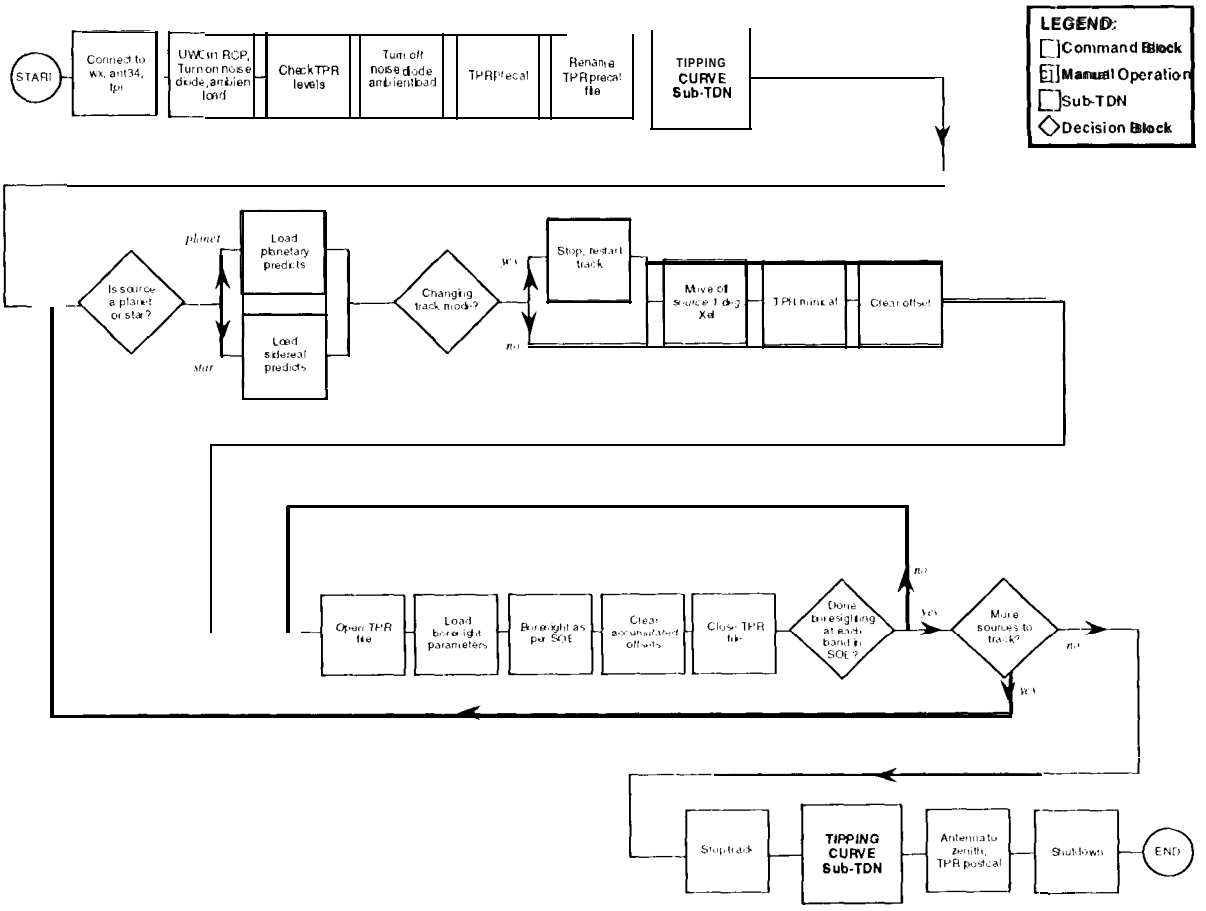


Figure 13: Ka-band Antenna Pointing (KaAP) track Temporal Dependency Network

6.0 Knowledge Engineering

The goal-driven automation system we have described requires an extensive knowledge base to store data about the domain and as such requires that knowledge engineering be performed in order to acquire and encode the knowledge. In this section we describe the level and types of effort required to acquire, encode and validate the knowledge base used in this demonstration. The levels of effort can be summarized as follows and are in

system has successfully executed a command, a status message is sent to the block, and it dispatches the next command. Once all of the block's commands have been sent, the block sends a request to the Situation Manager to check whether the block's postconditions are satisfied -- the execution of the block is not considered successful unless the postconditions hold in the subsystems.

The EM tracks each stage of a block's execution -- it tracks whether the block's preconditions and postconditions are satisfied, and it also tracks the execution status of each of the block's commands. Much of this information is sent by the EM to the User Interface (UI), which graphic-ally depicts the execution status of the TDN to the user. The display uses color to indicate the status of each block and command. The UI indicates whether a block has been executed or not, whether it is currently executing or has been skipped or paused by the user, and whether there is an execution problem such as an unsatisfied precondition. The Operator interacts with the LMCOA by watching the execution of the TDN; the operator can pause or skip portions of the TDN, check the status of commands within individual blocks, and provide inputs where they are required. When part of a TDN fails, the LMCOA supports manual recovery by the operator by highlighting the point of failure and providing information about the preconditions or postconditions that failed to be satisfied.

The Situation Manager (SM) is responsible for tracking the evolution of the monitor and control subsystems over the duration of the track. In order to perform this task it embodies significant knowledge of how to query the subsystems to determine the information about their state and how to interpret responses. This knowledge is represented as a set of inference rule within the SM knowledge base.

l'here are several obvious drawbacks to operating the M&C system manually. Certain DSN operations require continuous attendance by an operator over long periods of time, and some operations are highly repetitive and require large amounts of data entry. For instance, it is not unusual to conduct a Ka-band Antenna Pointing (KaAP) track lasting eight hours. During a KaAP track the procedure called 'Load Sidereal Predicts' is repeated many times (see the KaAP TDN in Figure 13 for an end-to-end view of the track); the Load Sidereal Predicts procedure requires inputs by the operator each time it is conducted. We estimate that an eight hour KaAP track

Acquire the knowledge	36 days
Background information	14
Reuse blocks from other TDNs	4
Interview experts	11
Produce TDN documentation	7
Encode the knowledge	45 days
Project SOE Parser	10
Monitor & Control system extended	11
Adapt LMCOA to Voyager TDN	15
Write, revise TDN hat file	4
Decomposition rules, syntax editor	5
Validate the knowledge base	127 days
TDN, pre and post conditions	51
Monitor and Control system extensions	19
LMCOA adapted for Voyager TDN	24
Project SOE Parser	7
Goldstone testing	24
DPLAN	2

units of 8-hour workdays. It took about 36 workdays to acquire the knowledge, almost the same amount of time to encode the knowledge, but three times as much time to validate the knowledge base. 'T'able 1 summarizes the knowledge engineering effort. Details about each phase of this effort are described below.

Most of the time acquiring pre and postconditions occurred during the validation phase, due to the way the procedural knowledge is captured from operations personnel. Therefore details about acquiring pre and postconditions are presented later in the validation section.

11 days were spent interviewing experts in the areas of DSN Operations in general, 1)SS13 Operations for a spacecraft telemetry downlink pass, subsystem monitor and control, and project SOE definition. in addition, members of the Voyager project were interviewed in order to obtain parameters specific to Voyager that were required for parametrizing the TDN. 7 more days were spent generating graphical and written documentation of the TDN and **4 days** were spent participating in status meetings.

6.2 Encoding the Knowledge

The next step in the knowledge engineering process is to encode the knowledge into the right format in order for it to be machine readable. The majority of knowledge representations were designed and already used in previous efforts, due to the existing LMCOA, automated Monitor and Control System, and planner. In this effort the majority of the encoding time was spent putting the acquired knowledge into a known format. Encoding the knowledge covers writing the TDN flat file, which specifics the Voyager TDN and is loaded by the LMCOA, adapting the LMCOA to Voyager, writing the parser for the project SOE, extending the DSS13 Monitor and Control system to automate required subsystems, and developing DPLAN. Some highlights of the encoding effort arc noted below.

The DSS13 Monitor and Control system was expanded to include 2 new subsystems which were not required in other TDNs developed for the LMCOA. This included developing simple subsystem simulators for testing purposes. The integration of 1'1'13 into the Monitor and Control environment took the most amount of time. The changes made to the LMCOA for the new Voyager TDN took 15 days. Some of this time was spent extending the definition of the LSOE (LMCOA SOE) to account for new items in the Voyager SOE. As other types of tracks and more subsystems are handled by the LMCOA, the definition of the LSOE will broaden to i a-clude operations (e.g. boresight, modulation index changes) required by different tracks.

Table 1: Knowledge engineering effort

6.1 Acquiring the Knowledge

The majority of knowledge represented in the system consists of the project SOE definition, the DPLAN knowledge base (used in generating the TDN), the TDN blocks themselves, the parameterization of the TDN blocks by both DPLAN ant] LMCOA, and the directives which make up the TDN blocks used in the demonstration. Much of this information comes from subsystem knowledge, for example, in defining the TDN blocks, identifying anti understanding the subsystem directives within the blocks, and knowing how and when to parameterize subsystem directives within the blocks. Five subsystems were utilized for this TDN, two of which had never been used by LMCOA, and one that required more development anti testing to incorporate it into the current TDN. Acquiring this knowledge took about 36 workdays.

The information was obtained by several methods including reviewing documents and learning about the existing software systems, interviewing experts familiar with a particular part of the domain, documenting the knowledge and participating in status meetings. 14 days were spent acquiring background information on DSN operations, SOEs, the Monitor and Control System at DSS13, and the LMCOA at DSS13.

A small but very valuable amount of time was spent reviewing the existing TDNs for DSS13. A significant amount of knowledge was directly extracted from both the KaBLE and KaAP TDNs for use in the Voyager TDN. The operational unit of a TDN, namely a block, has proven advantageous in our previous work on TDNs. The reuse of blocks between different operations procedures is one key advantage. 12 of the 22 blocks, or 55% of the blocks, in the Voyager TDN came directly from the KaBLE and KaAP TDNs. Our plans for the next generation 1. MCOA include the use of a relational database to store TDNs, blocks, and their contents. A block that can be used in more than one TDN needs to be represented only once in the database. Changes could be made to one or more of these reusable blocks, without having to make significant, if any, changes to each TDN requiring the block. Therefore, to answer the question put forth in the overview and objectives section, the effort to extend this demonstration to a different type of track can be greatly reduced, depending on the reuse of TDN blocks from existing TDNs.

6.3 Validating the Knowledge Base

This phase of the knowledge engineering required about three times the level of effort of either the acquisition or encoding phases. This phase includes on-site testing at Goldstone in the operational environment, testing and developing simulators in the DSN Advanced Systems 1.ab in Building 525, and validating the TDN, pre and postconditions and other software modules. The validation phase took 127 workdays.

Almost half of this time, 51 days, was spent validating the TDN and especially pre and postconditions on TDN blocks. validating the TDN involves making sure that the operations procedure is accurate. One reason this is a time consuming task is that different experts have different ways of performing a part of the procedure. Or, over time they revise or and/or refine the operations procedure. This has been a constant in our experience developing TDNs. It is difficult to come to consensus on a single way to perform an operations procedure. After a TDN is in place and can be executed by the LMCOA, the operators can "see" the procedure more clearly and refine it. In any case, these reasons all point out the need to be able to modify TDN blocks and pre and postconditions. The format of the operations procedure must be simple so that it will be easy for developers, and operations personnel, to understand and maintain these procedures.

Pre and postconditions were extremely time-consuming to validate. From our previous experience building TDNs, we have found that operations personnel do not usually think in terms of pre and postconditions. The need for pre and postconditions is much more obvious after an initial demonstration of the baseline TDN. At this time, the operators observe subsystem actions occurring automatically in the TDN and identify when some actions occur before sufficient subsystem states have been reached. These states are then implemented as preconditions, Another reason why operators are not familiar with the concept of preconditions is related to how they operate the manual monitor and control environment. In this environment, the operators often have a lot of equipment pre configured. A detailed question and answer session between TDN developer and operator proved useful for identifying what portions of the pre configuration are preconditions for existing blocks in the TDN. Postconditions were more difficult for the operations personnel and developers to determine.

The two main reasons for the large amount of time validating pre and postconditions are I) the complexity of pre and postconditions and 2) subsystem support for them. Pre and postconditions can be very complex and therefore difficult to encode to have the desired effect. For example, a pre or postcondition based on absolute or relative time can complicate the implementation of that condition. In an early version of the Voyager TDN, two TDN blocks occur in sequence, START RECORDING and then S"1'0}' RECORDING. The START recording block just tells the appropriate subsystem to start recording. The block then completes. The STOP RECORDING block doesn't execute until the appropriate time has been reached, according to the time in the SOE. Until this time, it appears that no blocks are executing in the LMCOA. That is true, however, the subsystems are recording data during this time. In order for the user interface to show that some activity was occurring, a postcondition was put on the START RECORDING block. According to this postcondition, the START RECORDING block will finish executing when the time has come to stop recording. Therefore, during the long data recording phase of the pass, the start recording block in the LMCOA remains in the executing state.

Subsystem support is also required in order to make use of pre and postconditions. In this and previous TDNs, one or more subsystems did not provide status information to the monitor and control system that needed to be checked by a pre or postcondition. For example, in the Voyager TDN, files were downloaded to the SDR (Station Data Recorder). We had to modify this subsystem to send a "finished downloading" status back to Monitor and Control so that a postcondition could automatically test when downloading was co mpleted.

Extensions to the monitor and control system took 19 days to complete. The majority of this time was spent incorporating TP13, the receiver, into the Monitor and Control system. New variables were added to the subsystem and communications problems had to be overcome. This time also included writing subsystem simulators for testing Monitor and Control and the LMCOA in the 525 lab.

A significant amount of time, 24 days, was spent at 1)SS 13. Extensive testing and debugging was performed at Goldstone that could not be performed in the 525 lab environment. Much of this time was spent incorporating

'1"}'13 into the DSS13 Monitor and Control environment. Some of the difficulties encountered resulted in modifying sections of the knowledge base related to the Monitor and Control system as well as the LMCOA.

Finally, here are the levels of effort in debugging other software modules. Debugging the SOE Parser took 7 days and debugging DPLAN took 2 days. Another 24 days were required to debug the Voyager version of the LMCOA.

7.0 Results

in February 1995 a comprehensive demonstration was conducted to validate the concept of integrating and using the Al software described in the preceding paragraphs to track a spacecraft with the DSN [7, 8]. in the demonstration, DPLAN generated the TDN shown in Figure 4 for a Voyager Telemetry Downlink track using the equipment configuration at Deep Space Station 13 (1) SS-13) in Goldstone, California, which included a 34-meter beam-wave guide (BWG) antenna and a telemetry processor. The TDN generated by DPLAN was successfully executed by LMCOA--a communications link was established with Voyager and the 34-meter BWG antenna tracked the spacecraft, with minimal human control. As a result of this demonstration, DPLAN anti the concepts implemented in LMCOA are currently being transferred and implemented in the Network Control Project (NCP), which will replace two major DSN subsystems--the Monitor and Control (M&C) subsystem and the Network Planning and }'reparation (NPP) subsystem. While there are many outstanding research areas preventing complete end-to-end automation of the DSN (see [8] for a more detailed description of some of these), the concepts and technology demonstrated to date represent a significant step towards automation of routine DSN tracking services.

8.0 Discussion

in demonstrating the automation at DSS-13, several important lessons were learned. First, the validation of the knowledge base was a much more arduous task than expected. Considerable change occured in the LMCOA

knowledge base as a result of extensive testing on the actual operational equipment. This testing revealed the extreme difficulty of direct encoding of the low level procedural knowledge to execute DSN tracks. This valuable lesson has significant impact on the eventual automation of DSN operations - it means that significant resources need to be devoted to validating the low level antenna operations procedures and adequate time for testing and validation of these procedures is a must. In retrospect, given the complexity of the individual subsystems this should not have been such a great surprise.

A second important lesson was that modifying, the planner knowledge base to account for changes in the TDNs was not overly difficult. This is shown by the relatively small amounts of effort at knowledge engineering for the planner component relative to the execution component. This is clue to the fact that the planner was not as severely challenged since it was only required to generate a few types of TDNs. In a more complete test, it would have to generate a wide range of tracking TDNs (o support a range of spacecraft for different types of tracks.

A third lesson regarding the antenna operations domain is that the majority of the interactions and complexity is at the lower level. Designing the blocks (i.e. determining the low level directives to achieve and verify low level subsystem states) was much more complicated (and knowledge anti labor intensive) than composing the blocks themselves. Because of this complexity, further efforts in this area have heavily utilized smart tools for representing and reasoning about subsystems (such as finite state diagrams, and software engineering and analyis tools).

9.0 Conclusions

This paper has described the the application of Artificial Intelligence techniques for plan generation, plan execution, and plan monitoring to automate a Deep Space Communication Station. This automation enables a Communication station to respond to a set of tracking goals by appropriately reconfiguring the communications hardware and software to provide the requested communications services. In particular this paper has described:

(1) the overall automation architecture, (2) the plan generation and execution monitoring AI technologies used and implemented software components, and (3) the knowledge engineering process and effort required for automation. These technologies are currently being transferred to the operational Deep Space Network stations.

Acknowledgments

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This demonstration was funded by the DSN Advanced Technology Program, NASA Code O, managed by Dr. Chad Edwards. Elements of the technology demonstrated were previously funded by the operations/Artificial Intelligence Program, NASA Code X, managed by Dr. Mel Montemerlo. We would also like to thank Crista Smyth, Trish Santos, and Roland Bevan for their contributions to the DSS-13 demonstration.

References

[1] Deep Space Network, Jet Propulsion Laboratory Publication 400-517, April 1994.

[2] Final Report of the Services Fulfillment Re-engineering Team, JPL Interoffice Memorandum, March 14, 1995.

[3] K. Fayyad, R.W. Hill, Jr., and E.J. Wyatt. "Know'ledge Engineering for Temporal Dependency Networks as Operations Procedures." *Proceedings of A IAA Computing in Aerospace 9 Con ference*, 1993, San Di - ego, CA.

[4] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," *Proceedings oft/It Second International Conference on A I Planning Systems*, Chicago, IL, June 1994, pp. 249-254.

[5] J. S. Pemberthy and 1). S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, October 1992.

[6] S. A. Chien and H. B. Mortensen, "Automating Image Processing for Scientific Data Analysis of a Large Image Dat abase," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18 (8): pp. 854-859, August 1996.

[7] Hill, R. W., Jr., S. Chien, C. Smyth and K. Fayyad, "Planning for Deep Space Network." Proceedings of the 1995 AAAI Spring Symposium on integrated Planing Applications, Palo Alto, CA, 1995, AAAI Press.

[8] R. W. Hill, Jr., S. A. Chien, K. V. Fayyad, C. Smyth, T. Santos, anti R. Bevan, "Sequence of Events Driven Automation of the Deep Space Network," *Telecommunications and Data Acquisition 42-124,* October-December 1995.

[9] S. A. Chien, R. W. Hill Jr., X. Wang, T. Estlin, K. V. Fayyad, and H. B. Mortensen, "Why Real-world Planning is Difficult: A Tale of Two Applications," in *New Directions in AI Planning*, M. Ghallab and A. Milani, cd., 10S Press, Washington, DC, 1996, pp. 287-298.

[10] A. Lansky, "Action-based Planning," Proceedings of the Second International Conference on Al Planning Systems, Chicago, IL, June 1994, pp. 110-115.

[11] E. Kan, J. Rosas, and Q. Vu, "Operations Mission Planner - 26M Users Guide Modification 1.0", JPL Technical Document D- 10092, April 1996.

[12] S. A. Chien, "Knowledge Acquisition, Validation, and Maintenance in an Automated Planning System for image Processing," *Proceedings of the Tenth Workshop on Ktt[)}t'ledge Acquisition for Knowledge-based systems*, Banff, Canada, November 1996.